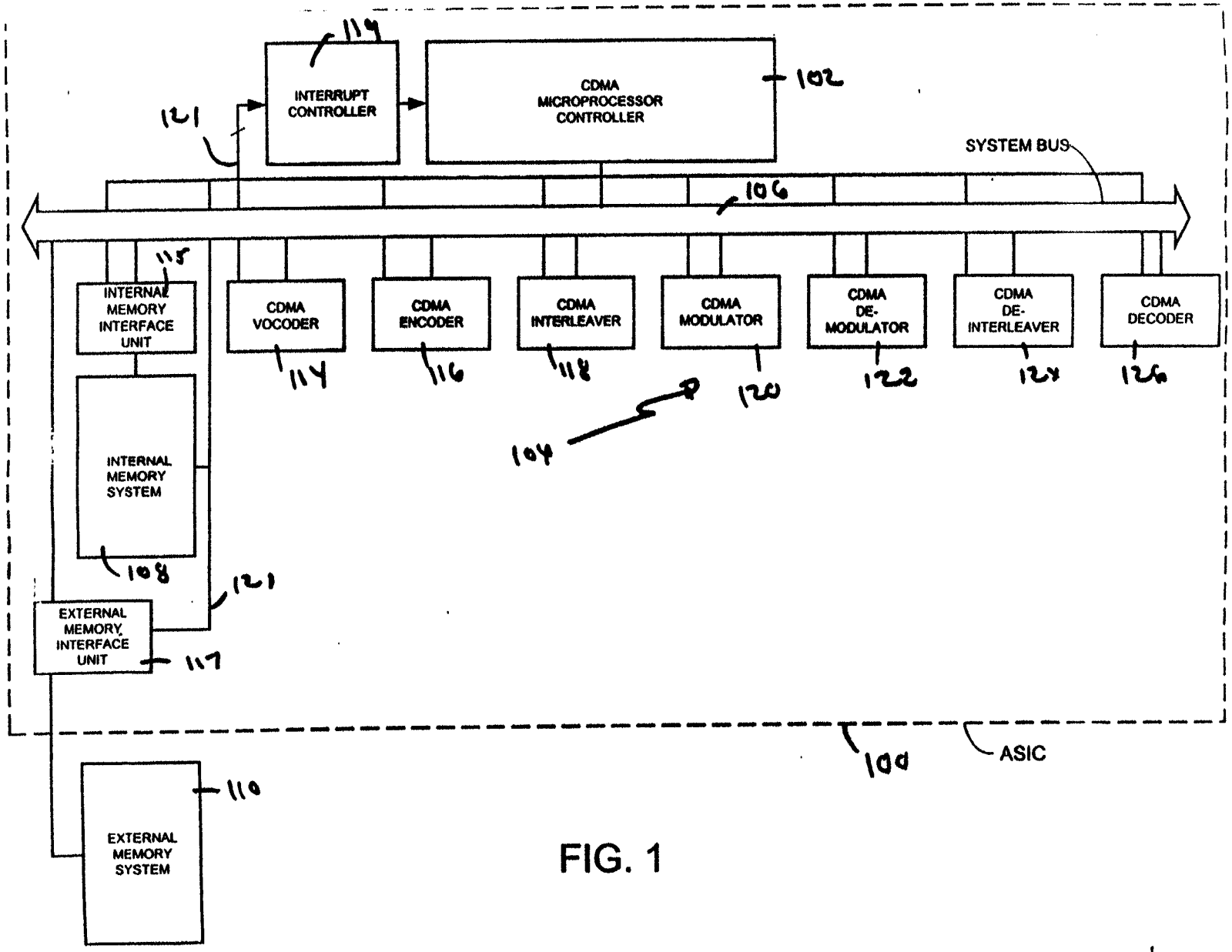
INTERRUPT CONTROLLER
CDMA MICROPROCESSOR CONTROLLER
SYSTEM BUS
INTERNAL MEMORY INTERFACE UNIT
CDMA VOCODER
CDMA ENCODER
CDMA INTERLEAVER
CDMA MODULATOR
CDMA DE-MODULATOR
CDMA DE-INTERLEAVER
CDMA DECODER
INTERNAL MEMORY SYSTEM
EXTERNAL MEMORY INTERFACE UNIT
EXTERNAL MEMORY SYSTEM
ASIC
114
102
121
106
115
108
117
110
104
116
118
120
122
124
126
100

FIG. 1

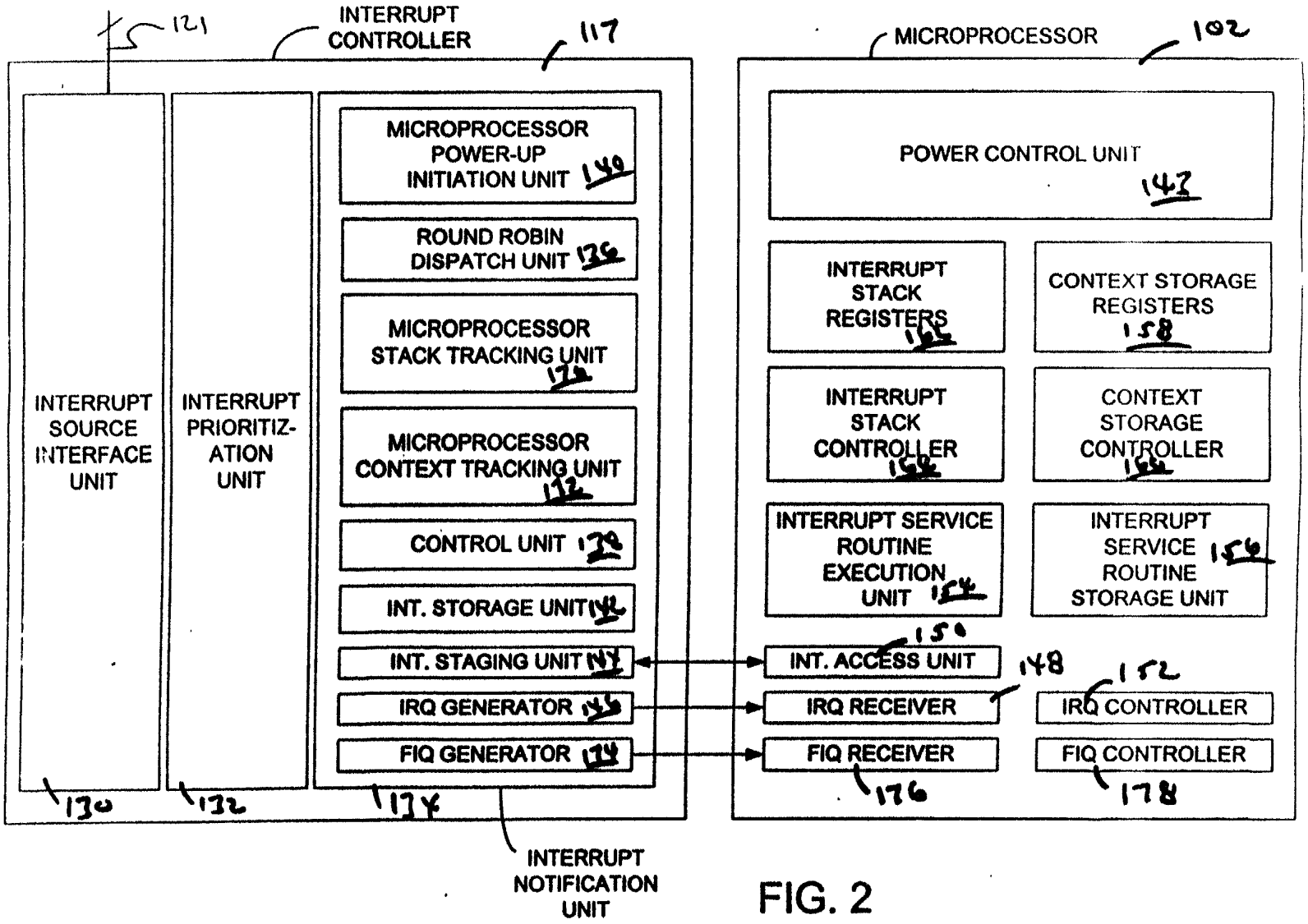
121
INTERRUPT CONTROLLER
117
MICROPROCESSOR
102
INTERRUPT SOURCE INTERFACE UNIT
INTERRUPT PRIORITIZ-ATION UNIT
MICROPROCESSOR POWER-UP INITIATION UNIT 140
ROUND ROBIN DISPATCH UNIT 126
MICROPROCESSOR STACK TRACKING UNIT 172
MICROPROCESSOR CONTEXT TRACKING UNIT 132
CONTROL UNIT 120
INT. STORAGE UNIT 142
INT. STAGING UNIT 144
IRQ GENERATOR 136
FIQ GENERATOR 134
POWER CONTROL UNIT 147
INTERRUPT STACK REGISTERS 152
CONTEXT STORAGE REGISTERS 158
INTERRUPT STACK CONTROLLER 153
CONTEXT STORAGE CONTROLLER 155
INTERRUPT SERVICE ROUTINE EXECUTION UNIT 154
INTERRUPT SERVICE ROUTINE STORAGE UNIT 156
150
INT. ACCESS UNIT
148
IRQ RECEIVER
152
IRQ CONTROLLER
FIQ RECEIVER
FIQ CONTROLLER
176
178
130
132
174
INTERRUPT NOTIFICATION UNIT

FIG. 2

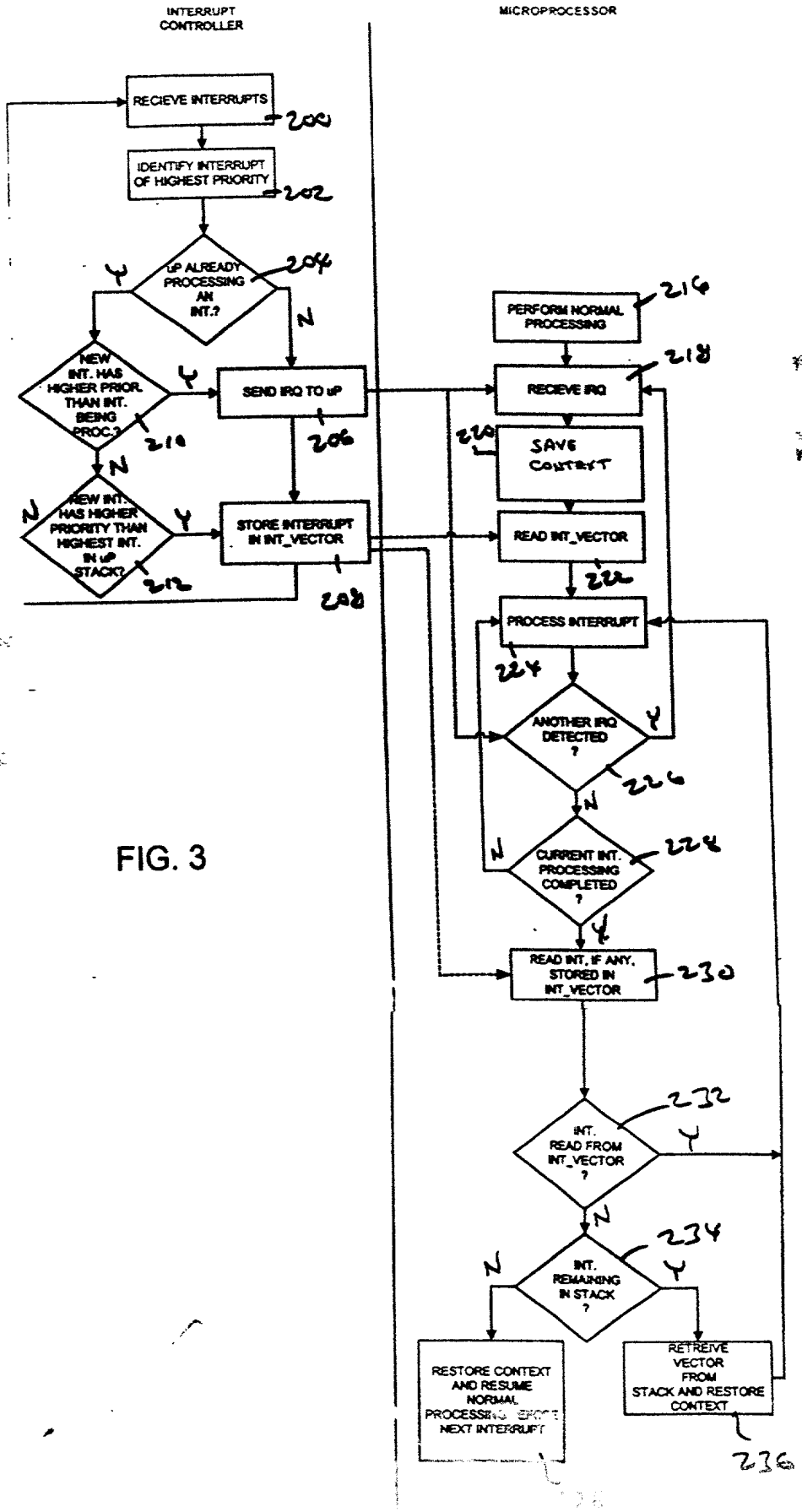
INTERRUPT CONTROLLER
MICROPROCESSOR
RECIEVE INTERRUPTS
200
IDENTIFY INTERRUPT OF HIGHEST PRIORITY
202
uP ALREADY PROCESSING AN INT.?
204
Y
N
NEW INT. HAS HIGHER PRIOR. THAN INT. BEING PROC.?
210
Y
N
SEND IRQ TO uP
206
NEW INT. HAS HIGHER PRIORITY THAN HIGHEST INT. IN uP STACK?
212
N
Y
STORE INTERRUPT IN INT_VECTOR
208
PERFORM NORMAL PROCESSING
216
RECIEVE IRQ
218
SAVE CONTEXT
220
READ INT_VECTOR
222
PROCESS INTERRUPT
224
ANOTHER IRQ DETECTED ?
226
Y
N
CURRENT INT. PROCESSING COMPLETED ?
228
N
Y
READ INT, IF ANY, STORED IN INT_VECTOR
230
INT. READ FROM INT_VECTOR ?
232
Y
N
INT. REMAINING IN STACK ?
234
N
Y
RESTORE CONTEXT AND RESUME NORMAL PROCESSING FROM NEXT INTERRUPT
RETREIVE VECTOR FROM STACK AND RESTORE CONTEXT
236

FIG. 3

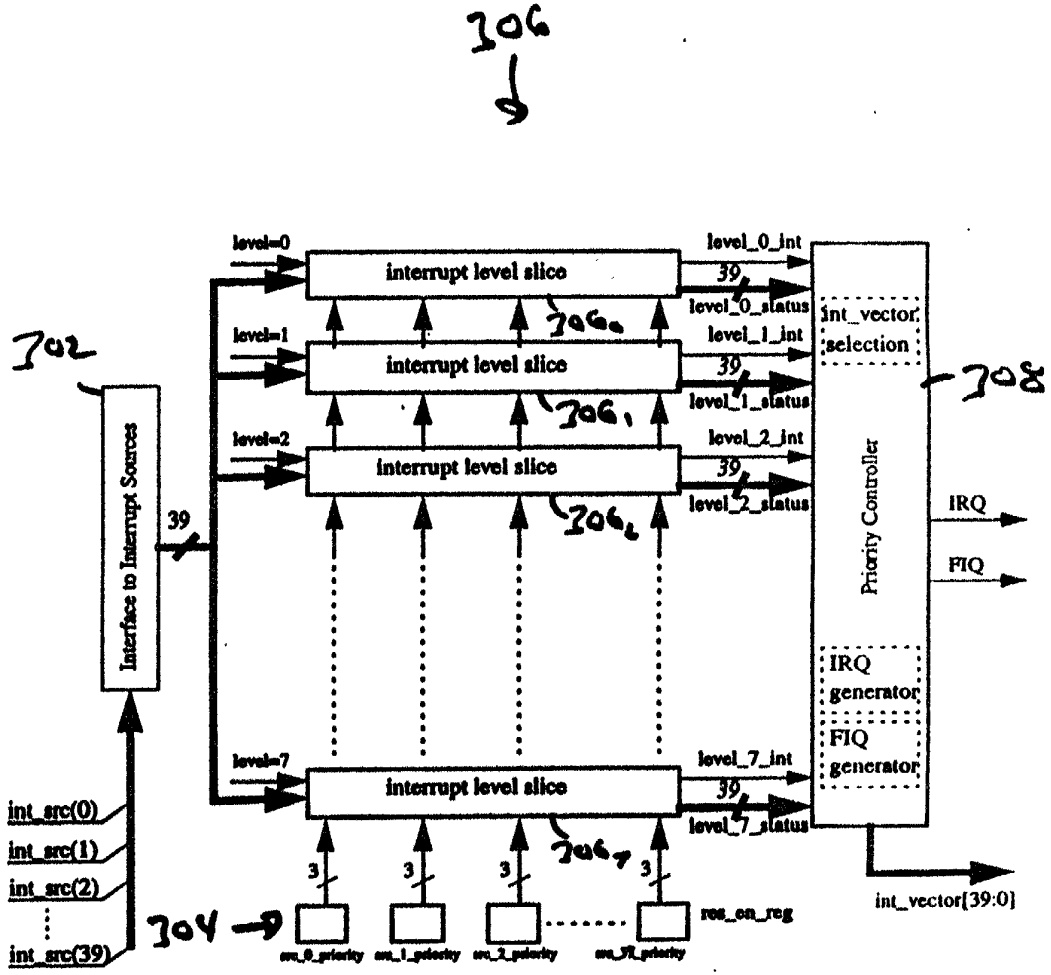
306
level=0
interrupt level slice
level_0_int
39
level_0_status
306₀
level=1
interrupt level slice
level_1_int
39
level_1_status
306₁
level=2
interrupt level slice
level_2_int
39
level_2_status
306₂
level=7
interrupt level slice
level_7_int
39
level_7_status
306₇
302
Interface to Interrupt Sources
39
int_src(0)
int_src(1)
int_src(2)
int_src(39)
304
3
res_en_reg
int_vector selection
Priority Controller
308
IRQ
FIQ
IRQ generator
FIQ generator
int_vector[39:0]

FIG. 4

300

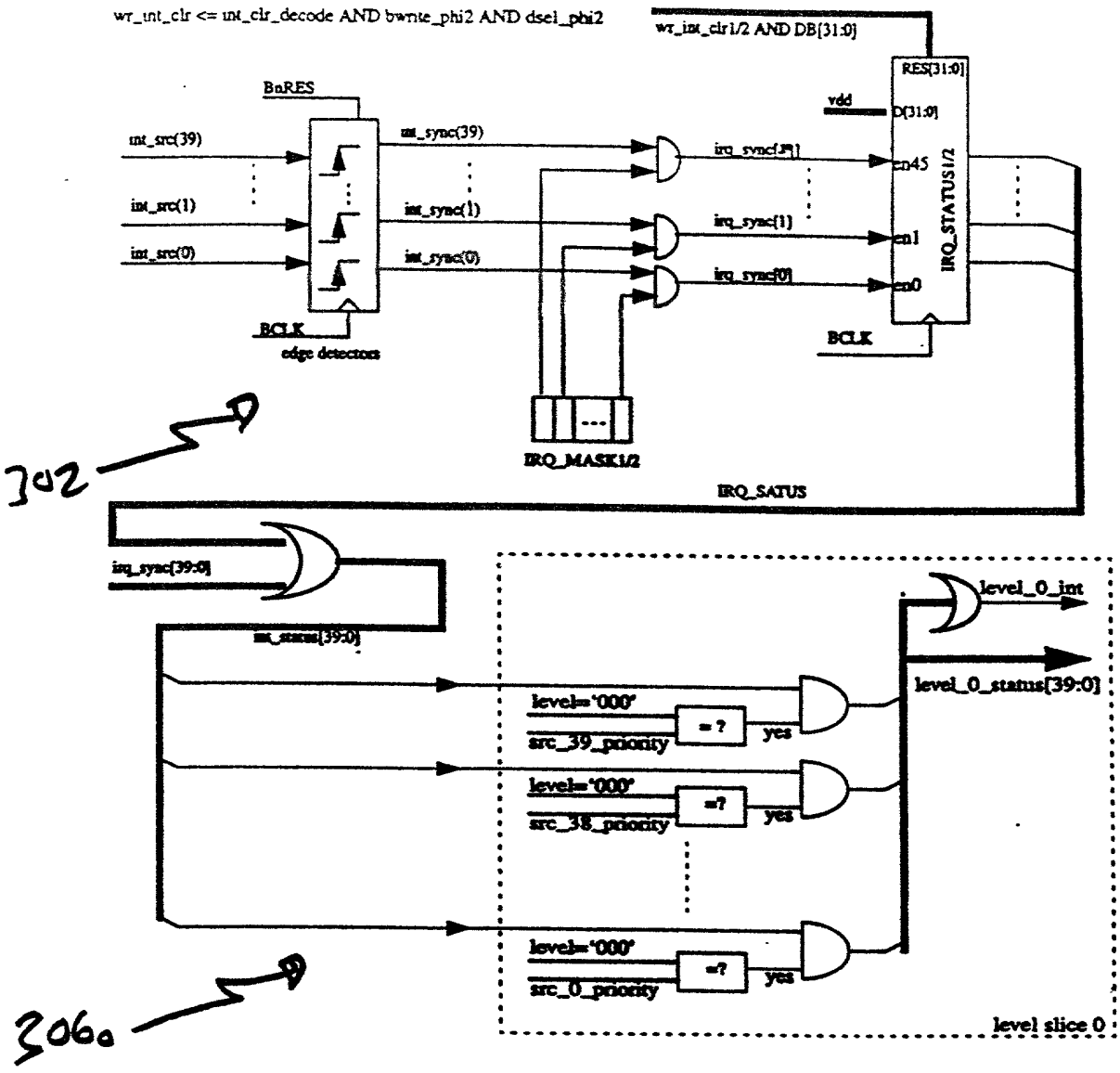
wr_int_clr <= int_clr_decode AND bwrite_phi2 AND dsel_phi2
wr_int_clr1/2 AND DB[31:0]
BnRES
int_src(39)
int_src(1)
int_src(0)
int_sync(39)
int_sync(1)
int_sync(0)
BCLK
edge detectors
irq_sync[39]
irq_sync[1]
irq_sync[0]
RES[31:0]
vdd
D[31:0]
en45
en1
en0
IRQ_STATUS1/2
BCLK
IRQ_MASK1/2
IRQ_SATUS
302
irq_sync[39:0]
int_status[39:0]
level_0_int
level_0_status[39:0]
level='000'
src_39_priority
=?
yes
level='000'
src_38_priority
=?
yes
level='000'
src_0_priority
=?
yes
level slice 0
306a

FIG. 5

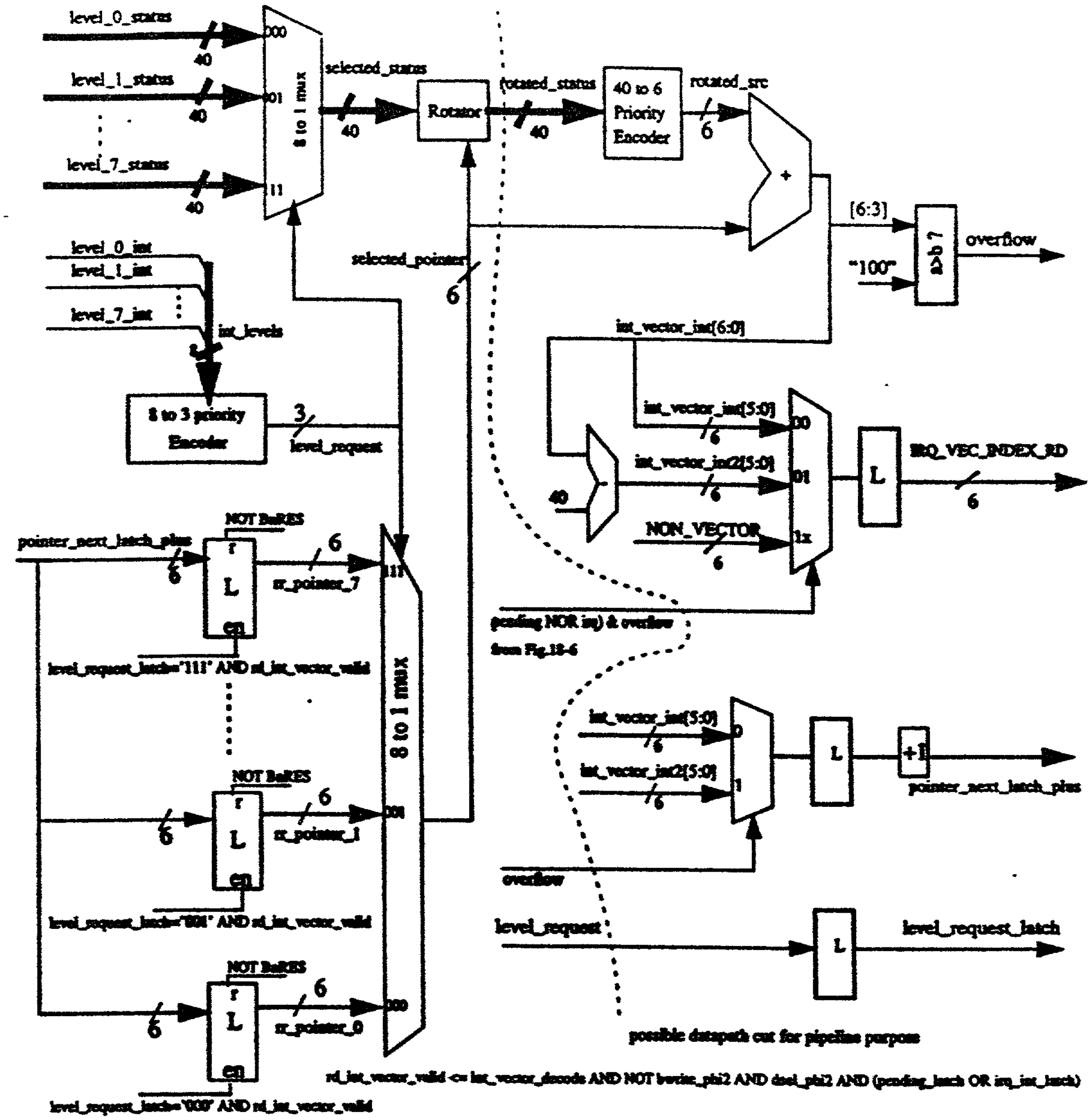
level_0_status
level_1_status
level_7_status
40
000
001
111
8 to 1 mux
selected_status
Rotator
rotated_status
40 to 6
Priority
Encoder
rotated_src
6
+
[6:3]
"100"
a>b ?
overflow
selected_pointer
level_0_int
level_1_int
level_7_int
int_levels
8
8 to 3 priority
Encoder
3
level_request
int_vector_int[6:0]
int_vector_int[5:0]
int_vector_int2[5:0]
NON_VECTOR
00
01
1x
L
IRQ_VEC_INDEX_RD
pointer_next_latch_plus
NOT BaRES
r
en
sr_pointer_7
sr_pointer_1
sr_pointer_0
level_request_latch='111' AND rd_int_vector_valid
level_request_latch='001' AND rd_int_vector_valid
level_request_latch='000' AND rd_int_vector_valid
(pending NOR irq) & overflow
from Fig.18-6
0
1
+1
level_request_latch
possible datapath cut for pipeline purpose

FIG. 6

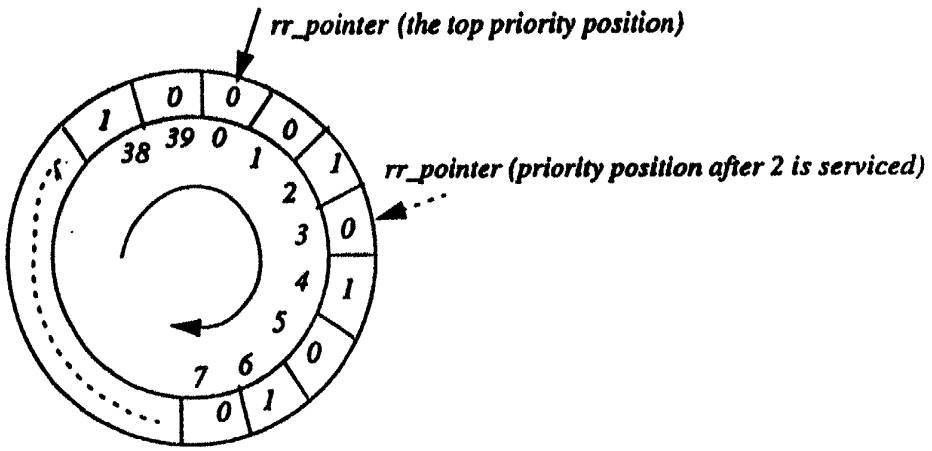
rr_pointer (the top priority position)
rr_pointer (priority position after 2 is serviced)
0
1
2
3
4
5
6
7
38
39
0
1

FIG.7

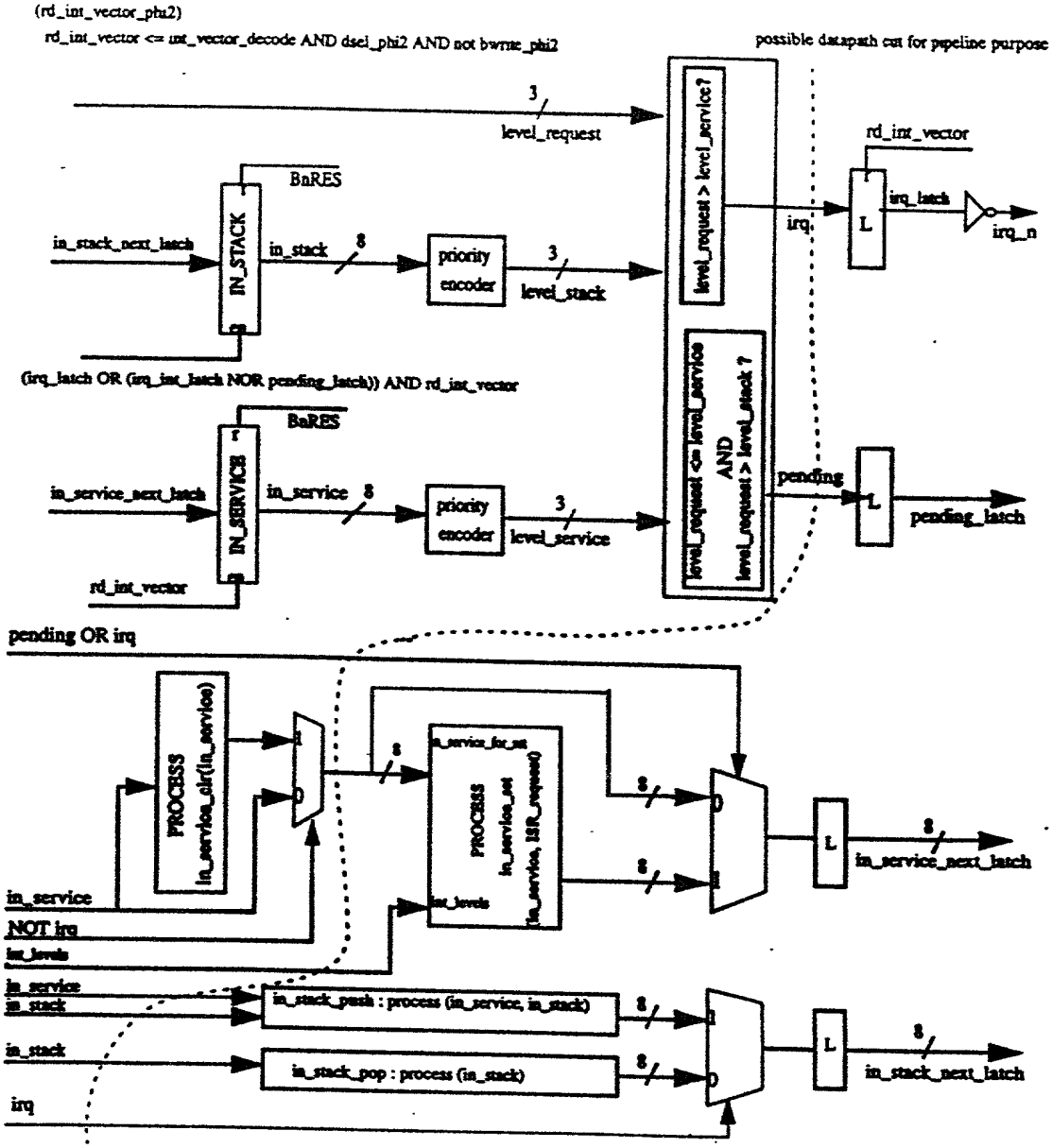
(rd_int_vector_phi2)
rd_int_vector <= int_vector_decode AND dsel_phi2 AND not bwrite_phi2
possible datapath cut for pipeline purpose
3
level_request
level_request > level_service ?
rd_int_vector
irq_latch
irq_n
irq
L
BaRES
IN_STACK
in_stack_next_latch
in_stack
8
priority encoder
3
level_stack
(irq_latch OR (irq_int_latch NOR pending_latch)) AND rd_int_vector
level_request <= level_service AND level_request > level_stack ?
pending
pending_latch
BaRES
IN_SERVICE
in_service_next_latch
in_service
priority encoder
level_service
rd_int_vector
pending OR irq
PROCESS
in_service_clr(in_service)
in_service_for_set
PROCESS
in_service_set
(in_service, ISR_request)
int_levels
in_service_next_latch
in_service
NOT irq
int_levels
in_service
in_stack
in_stack_push : process (in_service, in_stack)
in_stack
in_stack_pop : process (in_stack)
in_stack_next_latch
irq

FIG. 8

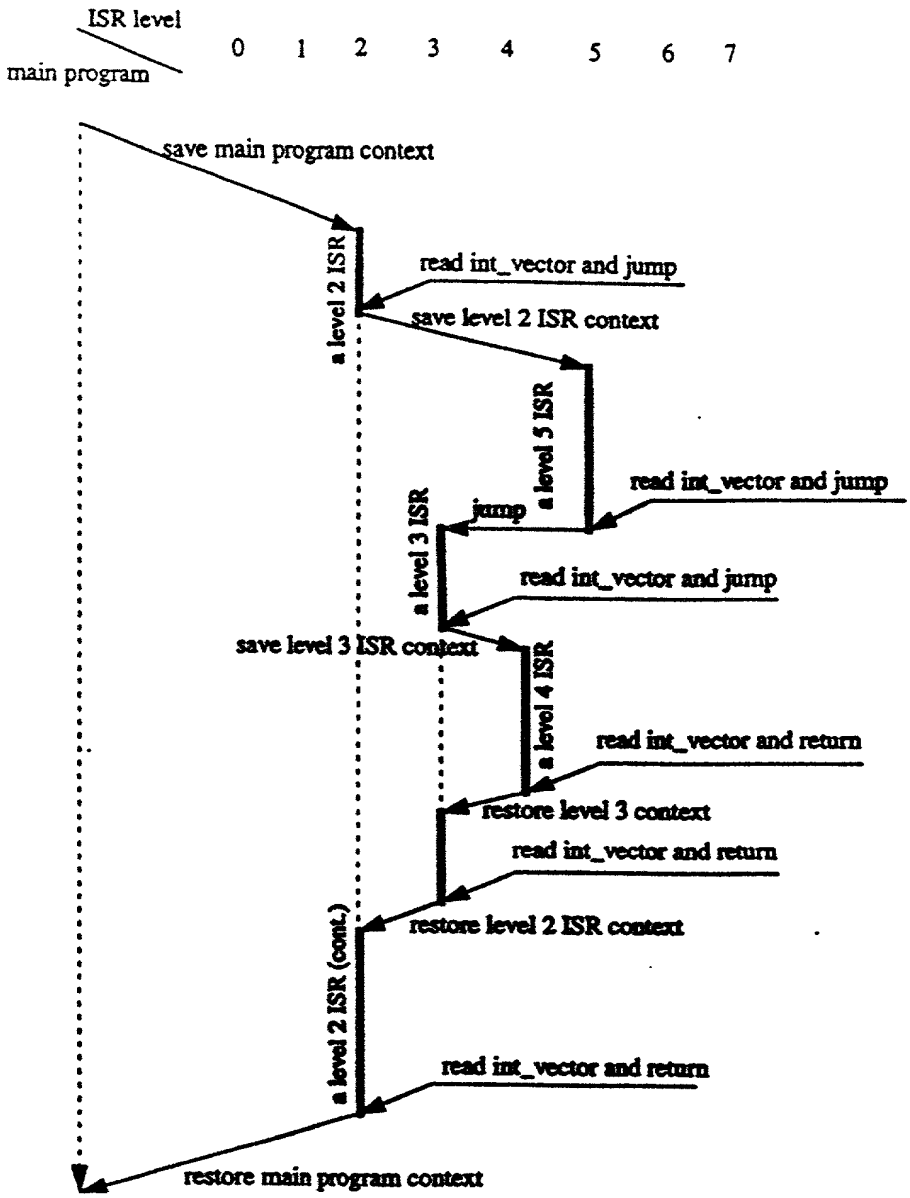
ISR level
0 1 2 3 4 5 6 7
main program
save main program context
a level 2 ISR
read int_vector and jump
save level 2 ISR context
a level 5 ISR
read int_vector and jump
jump
a level 3 ISR
read int_vector and jump
save level 3 ISR context
a level 4 ISR
read int_vector and return
restore level 3 context
read int_vector and return
restore level 2 ISR context
a level 2 ISR (cont.)
read int_vector and return
restore main program context

FIG. 9

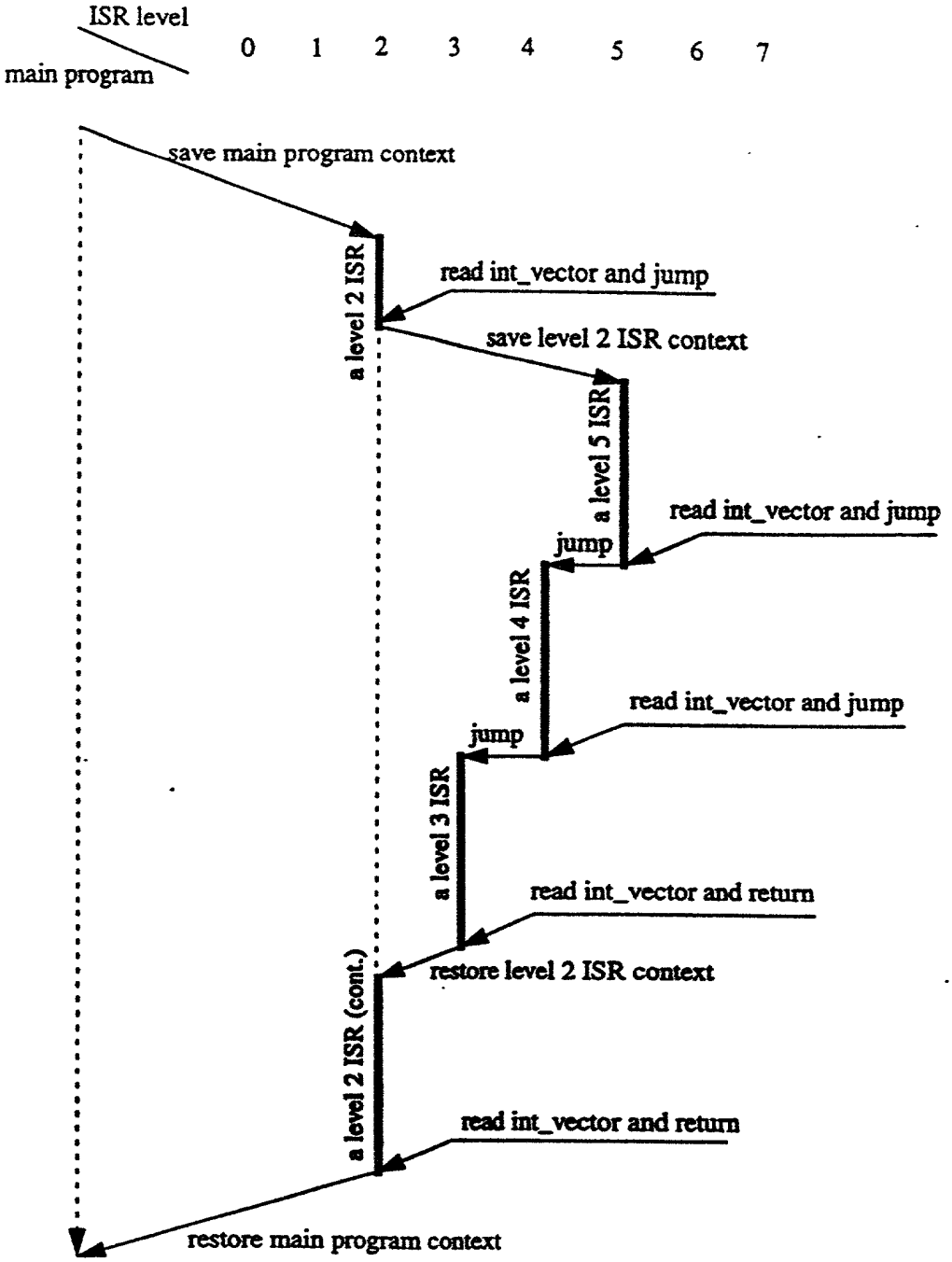
ISR level
0
1
2
3
4
5
6
7
main program
save main program context
a level 2 ISR
read int_vector and jump
save level 2 ISR context
a level 5 ISR
read int_vector and jump
jump
a level 4 ISR
read int_vector and jump
jump
a level 3 ISR
read int_vector and return
restore level 2 ISR context
a level 2 ISR (cont.)
read int_vector and return
restore main program context

FIG. 10

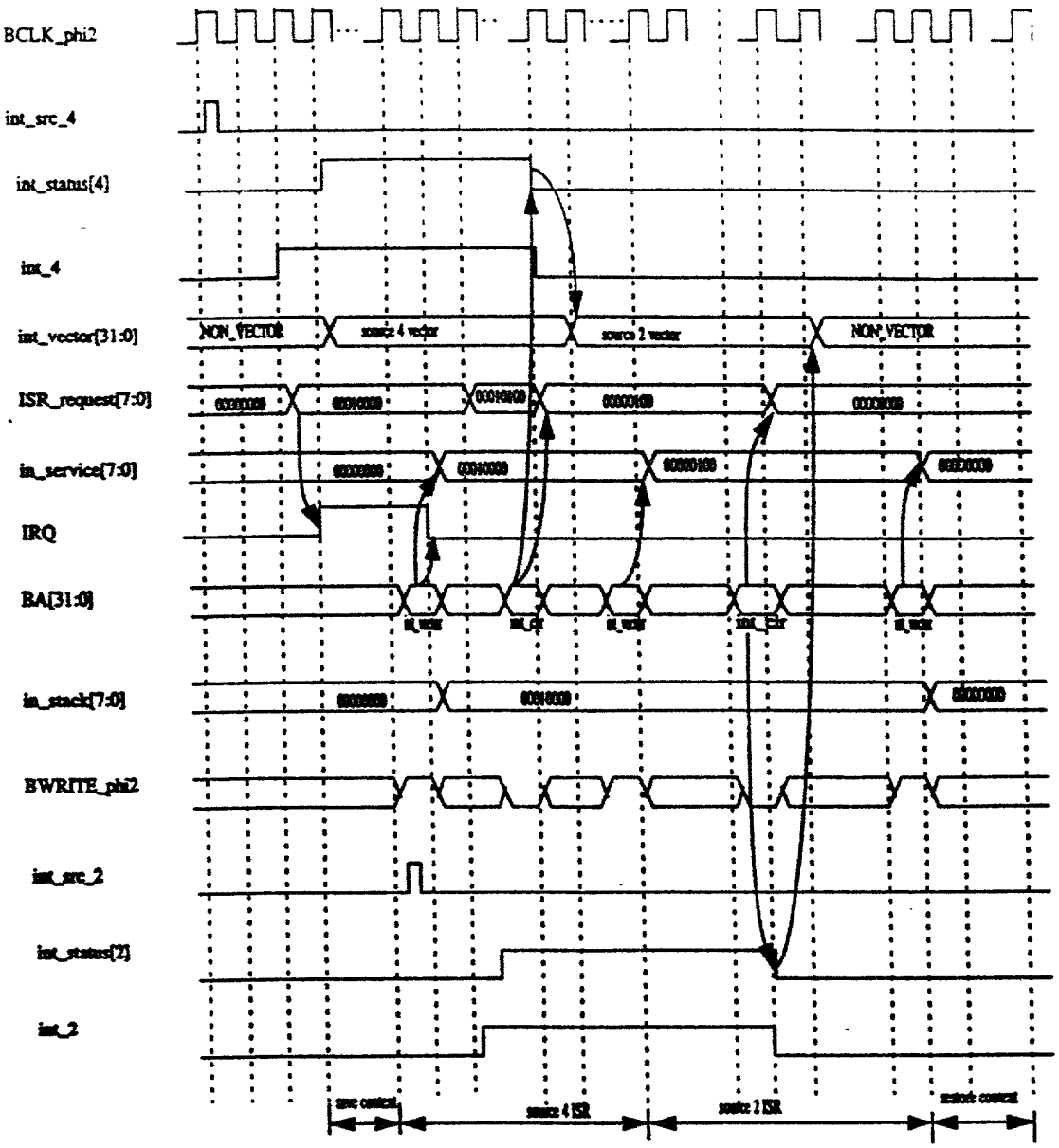
BCLK_phi2
int_src_4
int_status[4]
int_4
int_vector[31:0]
ISR_request[7:0]
in_service[7:0]
IRQ
BA[31:0]
in_stack[7:0]
BWRITE_phi2
int_src_2
int_status[2]
int_2
NON_VECTOR
source 4 vector
source 2 vector
NON_VECTOR
source 4 ISR
source 2 ISR

FIG. 11

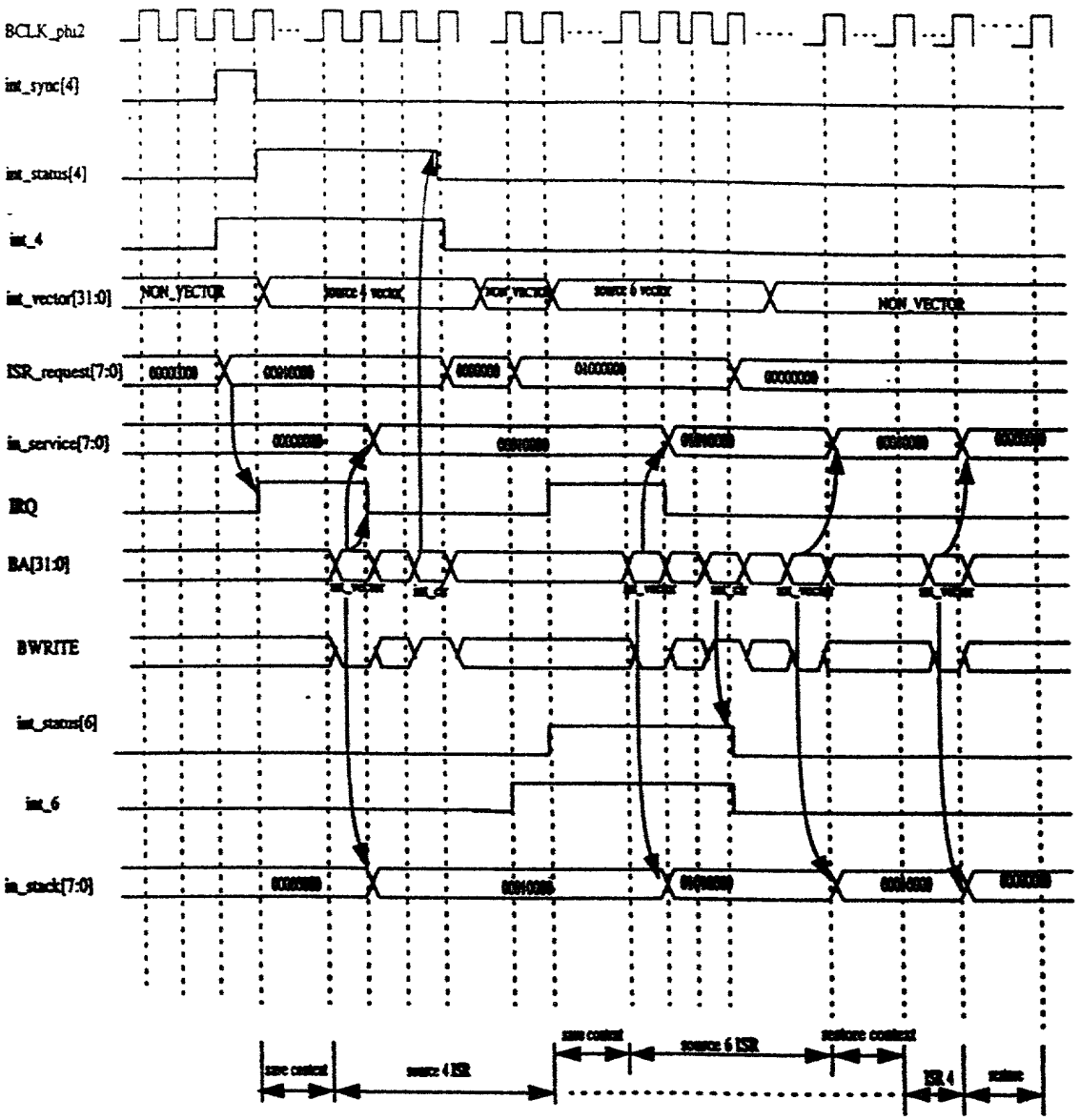
BCLK_ph2
int_sync[4]
int_status[4]
int_4
int_vector[31:0]
NON_VECTOR
source 4 vector
source 6 vector
NON_VECTOR
ISR_request[7:0]
in_service[7:0]
IRQ
BA[31:0]
BWRITE
int_status[6]
int_6
in_stack[7:0]
save context
source 4 ISR
save context
source 6 ISR
restore context
ISR 4
restore

FIG. 12

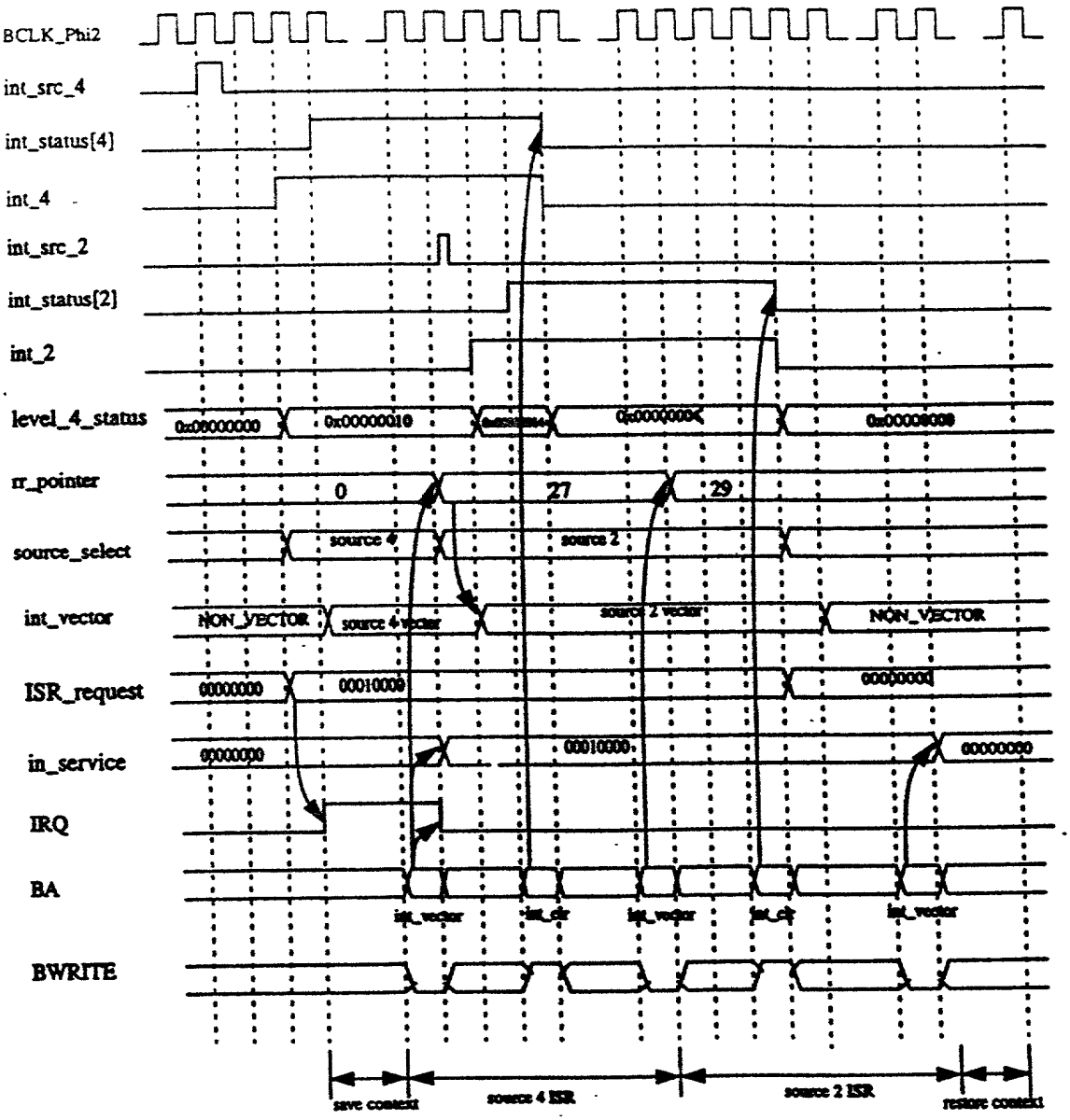
BCLK_Phi2
int_src_4
int_status[4]
int_4
int_src_2
int_status[2]
int_2
level_4_status
0x00000000
0x00000010
0x00000000
0x00000000
rr_pointer
0
27
29
source_select
source 4
source 2
int_vector
NON_VECTOR
source 4 vector
source 2 vector
NON_VECTOR
ISR_request
00000000
00010000
00000000
in_service
00000000
00010000
00000000
IRQ
BA
int_vector
int_clr
int_vector
int_clr
int_vector
BWRITE
save context
source 4 ISR
source 2 ISR
restore context

FIG. 13

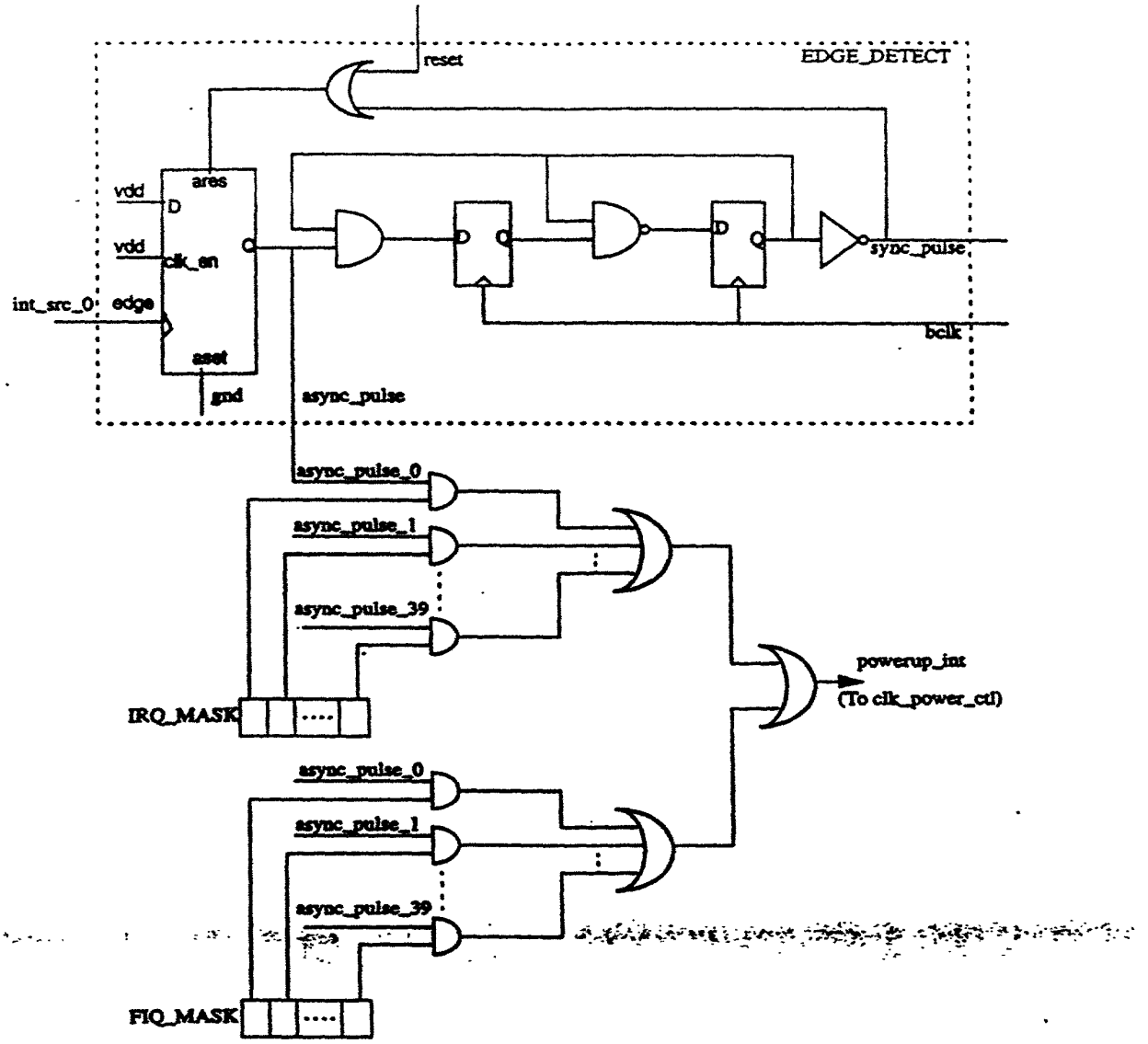
reset
EDGE_DETECT
vdd
D
ares
vdd
clk_en
int_src_0
edge
aset
gnd
async_pulse
sync_pulse
bclk
async_pulse_0
async_pulse_1
async_pulse_39
IRQ_MASK
powerup_int
(To clk_power_ctl)
async_pulse_0
async_pulse_1
async_pulse_39
FIQ_MASK

FIG. 14